Static Members

- We can declare a class basically contains two sections:
  - Declare instance variables
  - Declare the instance methods
- These are called instance members of a class.
- Because every time a the class is instantiated, a new copy of each of them is created.
- They are accessed using the objects.

- If we want to define a member that is common to all objects and accessed without using a particular object.
- That is a member belongs to the class as a whole rather than the objects created from the class.
- Such members are called static members.

- We can create static variables and static methods using the keyword static.
- For eg.

```
static int cout;
static int max(int x, int y);
```

Since these members are associated with the class itself rather than individual objects, the static variables and static methods are often referred to as class variables and class methods.

Static Variables or **class variables**

- Static variables are also called **class variables** because they can be accessed using class name.
- Static variables are used when we want to have a variable common to all instances of a class.
- Static variables occupy single location in the memory.
- These are accessible in both static and non-static methods, even non-static methods can change their values.
- Commonly the static variables are using in the case of we want to keep record about the number of running instances (objects) of a class.

Example for static variable

```
class Student
 {
   static int count=0; //static variable
   Student( )
    {//increment static variable
          count++;
   }
   void showCount( )
   {
          System.out.println("Number Of students : "+count);
   }
 }
class StaticDemo
{
public static void main(String  args[ ])
{
```

```
Student s1=new Student();
s1.showCount();
Student s2=new Student();
 s2.showCount();
Student s3=new Student();
Student s4=new Student();
s3.showCount();
s4.showCount();
Student.count=20;
s4.showCount();
 }
}
```

Output

```
Number Of students : 1
Number Of students : 2
Number Of students : 4
Number Of students : 4
Number Of students: 20
```

Static Methods/Class Methods

- When a method is declared with *static* keyword, it is known as static method.
- The most common example of a static method is *main( )* method.

Static Methods/Class Methods

- When a method is declared with *static* keyword, it is known as static method.
- The most common example of a static method is *main( )* method.
- Like static variables, static methods can be called without using the objects.

Example program for static method

```
class mathop
{
    static float mul(float x, float y)
    {
            return  x*y;
    }
    static float div(float a, float b)
    {
            return x/y;
    }
}
class staticmeth
{
    public static void main(String arg[])
    {
            float m=mathop.mul(14.5,10.2);
            float d=mathop.div(m,2.0);
            System.out.println("Product="+m);
            System.out.println("divide="+d);
    }
}
```

Static class in Java

- Java allows a class to be defined within another class. These are called **Nested Classes**.
- The class in which the nested class is defined is known as the Outer Class.
- Unlike top-level classes, Nested classes can be Static.
- Non-static nested classes are also known as Inner classes.
- Static classes in Java are allowed only for inner classes which are defined under some other class, as static outer class is not allowed which means that we can't use static keyword with outer class.

- The top level class cannot be static in java, to create a static class we must create a nested class and then make it static.

- Differences between Static and Non-static Nested Classes
  - A static nested class may be instantiated without instantiating its outer class.
  - Inner classes can access both static and non-static members of the outer class. A static class can access only the static members of the outer class.

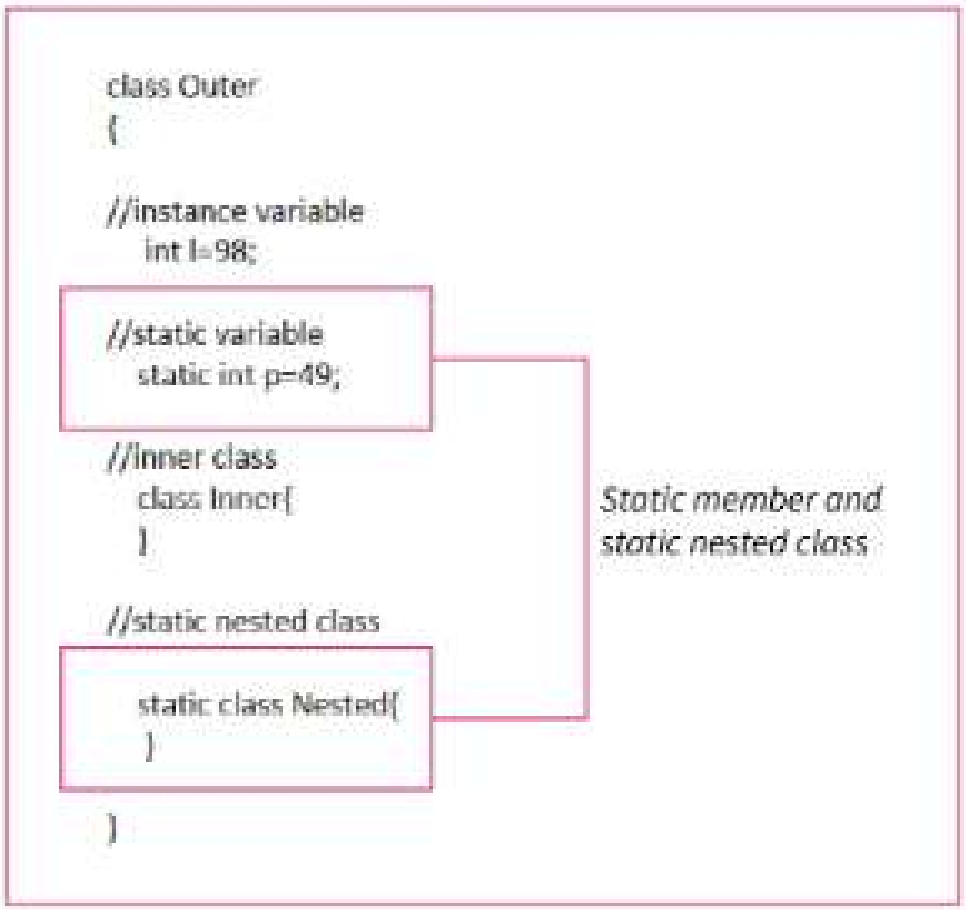
- A class can be declared static only if it is a nested class.
- It does not require any reference of the outer class.
- The property of the static class is that it does not allows us to access the non-static members of the outer class.

- **Inner class(a class within a class)**
- The classes that are non-static and nested are called inner classes.
- Note that we cannot create an instance of the inner class without creating an instance of the outer class.
- Without using the reference to the outer class instance, an instance of the inner class can access the members of its outer class.
- It makes the program simple and concise.

- **Outer Class**
- The class in which nested class is defined is called outer class.

- **Nested Class**
- Java allows us to define a class within a class that is known as a nested class. It may be static or non-static.

- An instance of a static nested class can be created without the instance of the outer class.
- The static member of the outer class can be accessed only by the static nested class.


class Outer
{
//instance variable
int i=98;
//static variable
static int p=49;
//Inner class
class Inner{
}
//static nested class
static class Nested{
}
}
Static member and
static nested class

Java Inner Classes- Example program

To access the inner class, create an object of the outer class, and then create an object of the inner class:

```
class OuterClass {
  int x = 10;

  class InnerClass {
    int y = 5;
  }
}

public class Main {
  public static void main(String[] args) {
    OuterClass myOuter = new OuterClass();
    OuterClass.InnerClass myInner = myOuter.new InnerClass();
    System.out.println(myInner.y + myOuter.x);
  }
}
```

Outputs
15

Static Inner Class

An inner class can also be static, which means that you can access it without creating an object of the outer class:

```
class OuterClass {
  int x = 10;

  static class InnerClass {
    int y = 5;
  }
}

public class Main {
  public static void main(String[] args) {
    OuterClass.InnerClass myInner = new OuterClass.InnerClass();
    System.out.println(myInner.y);
  }
}
```

Output

5

Example

```
class Outer {

  // static member of the outer class
  private static char grade = 'A';

  // Static class
  static class Nested {

    //non-static method
    public void fun() {
      // nested class can access the static members of the outer class
      System.out.println("Grade: " + grade);
    }
  }

  public static void main(String args[]) {
    Outer.Nested obj = new Outer.Nested(); //creating an object of nested
    // class without creating an object of the outer class.
    obj.fun();
  }
}
```

Output:

Grade: A

- The nested static class can access the static variables.
- Static nested classes do not have access to other members of the enclosing class which means it has no access to the instance variables (non-static variables) which are declared in the outer class.